Emmanuel Twum

Foundations of Programming Python

Randal Root

Assignment 06

11/4/2017

Reading and Writing Dictionary Items To/From a Text File Using a Class and Functions

The purpose of this paper is to describe how to create a python program using a ‘Class’ and ‘Functions’ which will load data from a text file and display it as a dictionary, add new items to the dictionary, remove existing items from the dictionary, and save data back to the text file. The program code is broken up into a section for data, processing, input/output. The program code assumes a text file called ToDo.txt is already created in the folder C:\\_PythonClass\.

The first section of the code is the data section (See Figure 1). The purpose of the data section is to define the variables and/or constants used in the program. The first variable defined is the variable ‘file’ which will be used later on in the code to open a text file in read mode and also write to the text file (See Figure 1). A variable for a blank table is defined next which will be appended with new rows (see Figure 1).

**Figure 1:** Data

#\*\*\*\*Data\*\*\*\*\*

#declare a variable file

file = ()

#create blank table

tblData = []

The next part of the code is the processing data. The first part of the processing section is the creation of the ‘class’ called ToDoList(), which will be the blueprint for running the program (See Figure 2). The ‘class’ is then broken out into five different functions. The first function, as shown in Figure 3, is to load the current data in the text file by reading the existing lines in the text file then taking the lines and converting them to dictionary rows. The rows are then appended to the table tblData (See Figure 3). The ‘@staticmethod’ defined before the function is used so no argument has to be passed to the method (see Figure 3). The second function, as shown in Figure 4, takes the tblData variable and shows what data is currently in it.

**Figure 2**: Class

class ToDoList():

#includes methods defined in the class

**Figure 3:** Load Current Data

@staticmethod

def LoadData():

"""reads text file then converts the current lines to dictionary rows in a table"""

#open the text file in read only mode

file = open("C:\\_PythonClass\ToDo.txt","r+")

#convert each row in the text file to a dictionary; append each dictionary into a table

for line in file:

strData = line

dicRow = {"Task":strData.split(",")[0],"Priority":strData.split(",")[1]}

tblData.append(dicRow)

**Figure 4:** Show Current Table Data

@staticmethod

def ShowCurrentData():

"""show current table data"""

return print(tblData)

The next function is for adding a new task/priority to the table tblData (see Figure 5). The function gets user input for the task and the priority which are then converted to a dictionary row. The new row is then appended to the existing data in the table.

**Figure 5:** Add Task to Table

@staticmethod

def AddItem():

"""add task to the table"""

while(True):

#get user input for task and priority

strTask = input("Please enter a task: ")

strPriority = input("Please enter the task's priority (low, medium, high): ")

#convert users inputs into a dictionary row

dicNewRow = {"Task":strTask.capitalize(), "Priority":strPriority.lower()+"\n"}

#append current table with new row

tblData.append(dicNewRow)

userinput = input("Would you like to continue inputting items? (Y/N) ")

if(userinput.lower() == 'y'):

continue

else:

break

The fourth function of the program is used to remove items from the table tblData based on user input (see Figure 6). The user inputs the task he or she would like to remove then the program lookups up the value in the task column of the dictionary rows. The task is then removed if it’s found in a row.

**Figure 6:** Remove Item from Table

@staticmethod

def RemoveItem():

"""Remove item from the table"""

boolTask = False

#get user input for removing a task

strRemoveTask = input("Input the task you would like to remove: ")

#look up user input in each row in the task column; delete row if found

for x in tblData:

if(strRemoveTask.capitalize() == x ["Task"]):

tblData.remove(x)

print("Task Deleted")

boolTask = True

if(boolTask == False):

print("Task not Found")

The last function is for saving the latest task information back to the text file (see Figure 7). The file is opened in write mode which overwrites the current text file with the new data. The values of the dictionary rows are then pulled from the table, converted to a list, and then saved to the text file as ‘task, priority’.

**Figure 7:** Save Data to File

@staticmethod

def SaveData():

"""save latest data to the text file"""

#overwrite current data in the file with new data

file = open("C:\\_PythonClass\ToDo.txt","w")

#pull values from the dictionary and turn them into a list; save to file

for x in tblData:

x.values()

this\_list = list(x.values())

file.write(this\_list[0] + "," + this\_list[1])

file.close()

The last part of the program is for loading the current data and the input/output section (see Figure 8). Prior to the user selecting an option, the text file is read and loaded using the LoadData() function. The user then inputs a number from the menu of options to choose what he or she would like to do. For each user option, an instance is created from the ‘class’ (x = ToDoList()). The instance then takes on each method from the class (ie x.ShowCurrentData()). The program then runs the function defined in the processing section of the code. This will continue to loop until the user inputs the number five to end the program.

**Figure 8:** Load Current Data and Input/Output

#load the current data in the text file

x = ToDoList()

x.LoadData()

#\*\*\*\*Input/Output\*\*\*\*

while(True):

print ("""

Menu of Options

1) Show current data

2) Add a new item.

3) Remove an existing item.

4) Save Data to File

5) Exit Program

""")

strChoice = str(input("Which option would you like to perform? Input [1 to 5] - "))

print()#adding a new line

**Figure 8:** Load Current Data and Input/Output (Continued)

# Step 3 -Show the current items in the table

if (strChoice == '1'):

x = ToDoList()

x.ShowCurrentData()

continue

# Step 4 - Add a new item to the list/Table

elif(strChoice == '2'):

x = ToDoList()

x.AddItem()

continue

# Step 5 - Remove a new item to the list/Table

elif(strChoice == '3'):

x = ToDoList()

x.RemoveItem()

continue

# Step 6 - Save tasks to the ToDo.txt file

elif(strChoice == '4'):

x = ToDoList()

x.SaveData()

continue

elif (strChoice == '5'):

break #and Exit the program